

1 **1. (currently amended)** A finite state model-based testing system
2 comprising:

3 a model generation engine configured to generate a model of a software
4 application, the model being generated from parameters that describe the software
5 application to be tested; and

6 a graphical user interface to enable user entry of the parameters for defining
7 to define the model; and

8 a test driver to initiate a test of the software application with a test sequence
9 of inputs generated from the model of the software application.

10 **2. (currently amended)** A finite state model-based testing system
11 as recited in claim 1, wherein the user interface enables a user to enter state
12 information and transition information about that describes the software
13 application, the transition information describing a next state of the software
14 application after an input has been applied to a current state of the software
15 application.

1 **3. (currently amended)** A finite state model-based testing system
2 as recited in claim 1, wherein the user interface enables a user to enter state
3 information ~~about that describes~~ the software application, the state information
4 comprising:

5 an operational mode of the software application, wherein the operational
6 mode is an attribute of a particular state of the software application;

7 at least one modal value ~~associated with the operational mode, wherein the~~
8 ~~modal value~~ that describes a behavior of the operational mode; and

9 an input of the software application.

10 **4. (currently amended)** A finite state model-based testing system
11 as recited in claim 1, wherein the user interface enables a user to enter transition
12 information ~~about that describes~~ the software application, the transition
13 information comprising:

14 a current state of the software application, the current state being associated
15 with an input of the software application; and

16 a next state ~~of the software application, the next state indicating that~~
17 ~~indicates~~ the state of the software application after the input has been applied to
18 the current state of the software application.

19 **5. (currently amended)** A finite state model-based testing system
20 as recited in claim 1, wherein the user interface comprises a model editor to enable
21 user entry of an operational mode, a modal value associated with the operational
22 mode, and an input of the software application ~~for defining to define~~ the model.

1 **6. (currently amended)** A finite state model-based testing system
2 as recited in claim 1, wherein the user interface comprises a rules editor to enable
3 user entry of an input of the software application, a current state of the software
4 application, and a next state of the software application indicating to indicate the
5 state of the software application after the input has been applied to the current state
6 of the software application for defining to define the model.

7
8 **7. (currently amended)** A finite state model-based testing system
9 as recited in claim 1, wherein the model of the software application is a state table,
10 the state table having at least one state table entry, and wherein:

11 a state table entry comprises:
12 (1) a current state of the software application;
13 (2) an input of the software application;
14 (3) a next state of the software application, the next state indicating
15 that indicates the state of the software application after the input has been
16 applied to the current state of the software application; and wherein
17 the model generation engine is further configured to evaluates evaluate the
18 current state of the software application to determine if an input of the software
19 application can be applied to the current state and in the event that the input can be
20 applied to the current state, writes a state table entry out to the state table.

1 **8. (currently amended)** A finite state model-based testing system
2 as recited in claim 1, wherein the user interface comprises a graph traversal menu
3 to enable a user to select a graph traversal program and generate a the test
4 sequence of inputs for the software application.

5
6 **9. (currently amended)** A finite state model-based testing system
7 as recited in claim 1, further comprising a graph traversal program to generate a
8 the test sequence of inputs for the software application; where the test sequence of
9 inputs are generated from the model of the software application with the graph
10 traversal program.

11
12 **10. (currently amended)** A finite state model-based testing system
13 as recited in claim 1, wherein the user interface comprises a test execution menu to
14 enable a user to select a the test driver program and initiate a the test of the
15 software application.

16
17 **11. (canceled)**

1 **12. (currently amended)** A user testing interface for testing a
2 software application, the user testing interface comprising:

3 a model editor to enable user entry of state information to define a model of
4 a the software application to be tested; and

5 a rules editor to enable user entry of transition information to further define
6 the model of the software application to be tested, the transition information
7 describing a next state of the software application after an input has been applied
8 to a current state of the software application; and

9 a model generation engine configured to generate the model of a software
10 application from the state information and the transition information.

11
12 **13. (currently amended)** A user testing interface as recited in
13 claim 12, wherein the user interface is further comprising a graphical user
14 interface that includes the model editor and the rules editor.

15
16 **14. (currently amended)** A user testing interface as recited in
17 claim 12, wherein the state information comprises:

18 an operational mode of the software application, wherein the operational
19 mode which is an attribute of a particular state of the software application;

20 at least one modal value associated with the operational mode, wherein the
21 modal value that describes a behavior of the operational mode; and

22 an input of the software application.

1 **15. (currently amended)** A user testing interface as recited in
2 claim 12, wherein the transition information comprises:

3 a current state of the software application, the current state being associated
4 with an input of the software application; and

5 a next state of the software application, ~~the next state indicating that~~
6 indicates the state of the software application after the input has been applied to
7 the current state of the software application.

8
9 **16. (currently amended)** A user testing interface as recited in
10 claim 12, wherein the model editor comprises:

11 an operational modes entry field to enable user entry of an operational
12 mode of the software application; and

13 an operational modes list field to display the operational mode.

14
15 **17. (currently amended)** A user testing interface as recited in
16 claim 12, wherein the model editor comprises:

17 a modal values entry field to enable user entry of a modal value associated
18 with an operational mode of the software application; and

19 a modal values list field to display the modal value.

1 **18. (currently amended)** A user testing interface as recited in
2 claim 12, wherein the model editor comprises:

3 an inputs entry field to enable user entry of an input of the software
4 application; and

5 an inputs list field to display the input of the software application.

6
7 **19. (currently amended)** A user testing interface as recited in
8 claim 12, wherein the rules editor comprises fields to display the state information
9 that can be entered using the model editor, the fields comprising:

10 inputs fields to display inputs of the software application, wherein the
11 inputs fields also enable user selection of an input of the software application;

12 operational modes fields to display operational modes of the software
13 application, wherein the operational modes fields also enable user selection of an
14 operational mode; and

15 modal values fields to display modal values associated with an operational
16 mode, wherein the modal values fields also enable user selection of a modal value.

1 **20. (currently amended)** A user testing interface as recited in
2 claim 19, wherein the rules editor enables user entry of the transition information,
3 and wherein:

4 the transition information comprises:

5 a current state of the software application, the current state being
6 associated with the input of the software application;

7 a next state ~~of the software application, the next state indicating that~~
8 indicates the state of the software application after the input has been
9 applied to the current state of the software application;

10 the rules editor comprises:

11 an inputs field to enable user entry of the input;

12 a current state operational mode field to enable user entry of the
13 operational mode as a current state operational mode;

14 a current state modal value field to enable user entry of the modal
15 value associated with the current state operational mode;

16 a next state operational mode field to enable user entry of the
17 operational mode as a next state operational mode; and

18 a next state modal value field to enable user entry of the modal value
19 associated with the next state operational mode.

1 **21. (currently amended)** A user testing interface as recited in
2 claim 12, wherein the model is a state table having at least one state table entry,
3 the state table entry having including a current state of the software application, an
4 input of the software application, and a next state of the software application, the
5 next state indicating that indicates the state of the software application after the
6 input has been applied to the current state of the software application; and wherein

7 the model editor enables user initiation of a the model generation engine to
8 generate the model of the software application, the model generation engine being
9 further configured to evaluate a current state of the software application to
10 determine if an input of the software application can be applied to the current state
11 and in the event that the input can be applied to the current state, writes write a
12 state table entry out to the state table.

13
14 **22. (currently amended)** A user testing interface as recited in
15 claim 12, wherein the model editor enables user initiation of a graph traversal
16 program to generate a test sequence of inputs for the software application.

17
18 **23. (currently amended)** A user testing interface as recited in
19 claim 12, wherein the user testing interface further comprises a graph traversal
20 menu to enable user selection of a graph traversal program to generate a test
21 sequence of inputs for the software application.

1 **24. (currently amended)** A user testing interface as recited in
2 claim 23, wherein the graph traversal menu comprises:

3 a graph traversal program field to enable user selection of the graph
4 traversal program;

5 a model file field to enable user selection of the model of the software
6 application to be tested; and

7 a test sequence file field to enable user entry of a memory storage location
8 for a test sequence file, the test sequence file containing the test sequence of inputs
9 for the software application.

10 **25. (currently amended)** A user testing interface as recited in
11 claim 12, wherein the model editor enables user initiation of a test driver program
12 to read a test sequence of inputs for the software application and apply the test
13 sequence of inputs to the software application.

15 **26. (currently amended)** A user testing interface as recited in
16 claim 12, wherein the user testing interface further comprises a test execution
17 menu to enable user selection of a test driver program to read a test sequence of
18 inputs for the software application and apply the test sequence of inputs to the
19 software application.

1 **27. (currently amended)** A user testing interface as recited in
2 claim 26, wherein the test execution menu comprises:

3 a test driver program field to enable user selection of the test driver
4 program;

5 a test sequence file field to enable user selection of a test sequence file
6 containing the test sequence of inputs for the software application ~~to-be-tested~~; and

7 a test monitoring interval field to enable user entry of a timing interval to
8 define how often the test driver program can be monitored to detect a failure of the
9 test driver program.

10 **28. (original)** A user interface to enable user entry of parameters to
11 define a model of a software application to be tested, the user interface
12 comprising:

13 an operational modes field to enable user entry of an operational mode of
14 the software application, the operational mode being an attribute of a particular
15 state of the software application; and

16 a modal values field to enable user entry of at least one modal value
17 associated with the operational mode, the modal value describing a behavior of the
18 operational mode.

19 **29. (original)** A user interface as recited in claim 28, further
20 comprising an input field to enable user entry of an input of the software
21 application.
22
23

1 **30. (original)** A user interface as recited in claim 29, further
2 comprising:

- 3 an operational modes list field to display the operational mode;
4 a modal values list field to display the modal value; and
5 an inputs list field to display the input of the software application.

6

7 **31. (original)** A user interface as recited in claim 28, wherein the
8 user interface enables user initiation of a graph traversal program to generate a test
9 sequence of inputs for the software application.

10

11 **32. (original)** A user interface as recited in claim 28, wherein the
12 user interface enables user initiation of a test driver program to read a test
13 sequence of inputs for the software application and apply the test sequence to the
14 software application.

15

16 **33. (original)** A user interface to enable user entry of parameters to
17 define a model of a software application to be tested, the user interface
18 comprising:

- 19 an inputs field to enable user entry of an input of the software application;
20 current state fields to enable user entry of a current state of the software
21 application, the current state being associated with the input; and
22 next state fields to enable user entry of a next state of the software
23 application, the next state indicating the state of the software application after the
24 input has been applied to the current state of the software application.

1 **34. (original)** A user interface as recited in claim 33, further
2 comprising a rules field to enable user entry of a rule to describe a transition of the
3 software application from a current state to a next state.

4

5 **35. (original)** A user interface as recited in claim 34, further
6 comprising a control to enable a user to disable a rule such that the model of the
7 software application will be defined without the rule.

8

9 **36. (original)** A user interface as recited in claim 33, wherein:

10 the current state fields include:

11 a current state operational mode field to enable user entry of a
12 current state operational mode of the software application;

13 a current state modal value field to enable user entry of at least one
14 current state modal value associated with the current state operational
15 mode;

16 the next state fields include:

17 a next state operational mode field to enable user entry of a next
18 state operational mode of the software application; and

19 a next state modal value field to enable user entry of at least one next
20 state modal value associated with the next state operational mode.

1 **37. (original)** A user interface as recited in claim 36, wherein:
2 the current state fields include:
3 a current state relational operator field to indicate the current state
4 modal value relation to the current state operational mode;
5 a current state concatenation operator field to indicate the relation
6 between a first current state rule criteria and a second current state rule
7 criteria.
8 the next state fields include:
9 a next state relational operator field to indicate the next state modal
10 value relation to the next state operational mode; and
11 a next state concatenation operator field to indicate the relation
12 between a first next state rule criteria and a second next state rule criteria.

13
14 **38-41. (canceled)**
15
16
17
18
19
20
21
22
23
24
25

1 **42. (currently amended)** A finite state model-based testing system

2 comprising:

3 a model editor to enable user entry of state information to define a model of
4 a software application to be tested;

5 a rules editor to enable user entry of transition information to further define
6 the model of the software application, the transition information describing a next
7 state of the software application after an input has been applied to a current state
8 of the software application;

9 a model generation engine to generate the model of the software
10 application, the model being generated from the state information and the
11 transition information;

12 a graph traversal menu to enable user selection of a graph traversal program
13 to generate a test sequence of inputs for the software application, the test sequence
14 of inputs being generated from the model of the software application; and

15 a test execution menu to enable user selection of a test driver program to
16 read the test sequence of inputs ~~for the software application~~ and apply the test
17 sequence of inputs to the software application.

1 **43. (original)** A finite state model-based testing system as recited in
2 claim 42, wherein the model editor comprises:

3 operational modes fields to enable user entry of an operational mode of the
4 software application;

5 modal values fields to enable user entry of a modal value associated with
6 the operational mode; and

7 inputs fields to enable user entry of an input of the software application.

8

9 **44. (original)** A finite state model-based testing system as recited in
10 claim 42, wherein the rules editor comprises fields to display the state information
11 that can be entered using the model editor, the fields comprising:

12 inputs fields to display inputs of the software application, wherein the
13 inputs fields also enable user selection of an input of the software application;

14 operational modes fields to display operational modes of the software
15 application, wherein the operational modes fields also enable user selection of an
16 operational mode; and

17 modal values fields to display modal values associated with an operational
18 mode, wherein the modal values fields also enable user selection of a modal value.

1 **45. (currently amended)** A finite state model-based testing system
2 as recited in claim 42, wherein the model is a state table having at least one state
3 table entry, the state table entry having including a current state of the software
4 application, an input of the software application, and a next state of the software
5 application; and wherein

6 the model editor enables user initiation of the model generation engine
7 which is configured to evaluate a current state of the software application to
8 determine if an input of the software application can be applied to the current state
9 and in the event that the input can be applied to the current state, writes write a
10 state table entry out to the state table.

11
12 **46. (original)** A finite state model-based testing system as recited in
13 claim 42, wherein the model editor facilitates user initiation of the rules editor.

14
15 **47. (original)** A finite state model-based testing system as recited in
16 claim 42, wherein the model editor facilitates user initiation of the graph traversal
17 menu.

18
19 **48. (original)** A finite state model-based testing system as recited in
20 claim 42, wherein the model editor facilitates user initiation of the test execution
21 menu.

1 **49. (currently amended)** A computer system comprising:

2 a processor;

3 a memory;

4 a user interface application stored in the memory and executable on the
5 processor to facilitate user definition of a finite-state model to test a software
6 application;

7 the user interface application having computer readable instructions to
8 display a graphical user interface; and

9 a model generation engine stored in memory and executable on the
10 processor to generate the finite-state model of the software application, the finite-
11 state model being generated from state information and transition information that
12 describes the software application.

13
14 **50. (currently amended)** A computer system as recited in claim
15 49, wherein the graphical user interface enables a user to enter the state
16 information and the transition information about that describes the software
17 application, the transition information describing a next state of the software
18 application after an input has been applied to a current state of the software
19 application.

20
21 **51. (currently amended)** A computer system as recited in claim
22 49, wherein the user interface comprises a model editor to enable user entry of
23 operational modes, modal values, and inputs of the software application to define
24 the finite-state model.

1 **52. (currently amended)** A computer system as recited in claim
2 49, wherein the user interface comprises a rules editor to enable user entry of an
3 input of the software application, a current state of the software application, and a
4 next state of the software application to define the finite-state model.

5
6 **53. (original)** A computer system as recited in claim 49, further
7 comprising at least one graph traversal program stored in the memory and
8 executable on the processor to generate a test sequence of inputs for the software
9 application, the user interface presenting a graph traversal menu to enable a user to
10 select the graph traversal program.

11
12 **54. (original)** A computer system as recited in claim 49, further
13 comprising at least one test driver program stored in the memory and executable
14 on the processor to execute a test sequence of application inputs on the software
15 application, the user interface presenting a test execution menu to enable a user to
16 select the test driver program.

17
18 **55. (original)** A computer system as recited in claim 49, further
19 comprising a data structure stored in the memory, the data structure comprising:

20 at least one mode data structure to hold an operational mode of a software
21 application and at least one modal value associated with the operational mode; and

22 at least one rule data structure to hold an input of the software application, a
23 current state of a software application, and a next state of the software application.

1 **56. (currently amended)** A method comprising:
2 presenting a graphical user interface that facilitates user entry of state
3 information and transition information ~~about~~ that describes a software application
4 to be tested; and
5 generating a model of the software application ~~using~~ from the state
6 information and the transition information; and
7 generating a test sequence of inputs for the software application from the
8 model of the software application.

9
10 **57. (currently amended)** A method as recited in claim 56, further
11 comprising: comprising presenting a graphical user interface that facilitates user
12 selection of a graph traversal program to generate the test sequence of inputs for
13 the software application; and
14 generating a test sequence of inputs for the software application.

15
16 **58. (currently amended)** A method as recited in claim 56, further
17 comprising:
18 presenting a graphical user interface that facilitates user selection of a test
19 driver program; and
20 executing a the test sequence of application inputs on the software
21 application with the test driver program.

1 **59. (original)** A method as recited in claim 56, further comprising
2 enabling a user to define a transition rule of the software application, wherein the
3 enabling comprises presenting a graphical user interface to facilitate user entry of
4 an input of the software application, a current state of the software application
5 associated with the input, and a next state of the software application.

6

7 **60. (original)** A method as recited in claim 56, further comprising:
8 presenting a graphical user interface that facilitates a user defining a
9 transition rule of the software application and disabling the transition rule; and
10 generating the model of the software application without incorporating the
11 disabled transition rule.

12

13 **61. (original)** A method as recited in claim 56, wherein generating a
14 model of the software application comprises:

15 evaluating a current state of the software application to determine if an
16 input of the software application can be applied to the current state; and
17 in the event that the input can be applied to the current state, writing a state
18 table entry out to a state table.

1 **62. (original)** A method as recited in claim 56, further comprising
2 enabling a user to define the state information, wherein the enabling comprises
3 presenting a graphical user interface to facilitate user entry of an operational mode
4 of the software application, at least one modal value associated with the
5 operational mode, and an input of the software application.

6

7 **63. (currently amended)** A method as recited in claim 56, further
8 comprising enabling a user to define the transition information, wherein the
9 enabling comprises presenting a graphical user interface to facilitate user entry of
10 a current state of the software application, ~~the current state being that is~~ associated
11 with an input of the software application, and user entry of a next state of the
12 ~~software application, the next state indicating that indicates~~ the state of the
13 software application after the input has been applied to the current state of the
14 software application.

15

16 **64. (original)** A computer-readable medium comprising computer
17 executable instructions that, when executed, direct a computing system to perform
18 the method of claim 56.

19
20
21
22
23
24
25

1 **65. (currently amended)** A method comprising:

2 presenting a user interface that facilitates user entry of state information and
3 transition information about a software application to be tested;

4 initiating, via the user interface, generation of a model of the software
5 application, the model being generated from the state information and the
6 transition information;

7 selecting, via the user interface, a graph traversal program that generates a
8 test sequence of inputs for the software application, the test sequence of inputs
9 being generated from the model of the software application; and

10 selecting, via the user interface, a test driver program that executes a test
11 sequence of application inputs on the software application.

12 **66. (original)** A computer-readable medium comprising computer
13 executable instructions that, when executed, direct a computing system to perform
14 the method of claim 65.

1 **67. (currently amended)** A method comprising:
2 receiving state information about a software application to be tested;
3 receiving transition information about the software application;
4 generating a model of the software application, the model being generated
5 from the state information and the transition information;
6 from the model, generating a test sequence of inputs for the software
7 application with a graph traversal program, the test sequence of inputs being
8 generated from the model of the software application; and
9 executing a the test sequence of application inputs on the software
10 application.

11
12 **68. (original)** A method as recited in claim 67, wherein generating a
13 model of the software application comprises:
14 evaluating a current state of the software application to determine if an
15 input of the software application can be applied to the current state; and
16 in the event that the input can be applied to the current state, writing a state
17 table entry out to a state table.

18
19 **69. (original)** A computer-readable medium comprising computer
20 executable instructions that, when executed, direct a computing system to perform
21 the method of claim 67.

1 **70. (currently amended)** A computer-readable medium
2 comprising computer executable instructions that, when executed, direct a
3 computing system to:

4 receive state information about a software application to be tested;
5 receive transition information about the software application;
6 generate a model of the software application from the state information and
7 the transition information;
8 ~~from the model,~~ generate a test sequence of inputs for the software
9 application with a graph traversal program, the test sequence of inputs being
10 generated from the model of the software application; and
11 execute a the test sequence of application inputs on the software
12 application.